第三节 数据存储之 JPA

数据库设计

users（主）

id	name	password
1	mercyblitz	`******`

addresses

id	address
10000	Shanghai
10001	Beijing

user_addresses（中间表）V2

id	user_id	address_id
100	10000	10000
101	10000	10001

id	address	user_id
10000	Shanghai	1

user_addresses（副） v1

id	address	user_id
10000	Shanghai	1
10001	Beijing	1

users (One)

user_addresses(Many)

User(id,name,password)

JPA 实体继承

users（主）

id	name	password
1	mercyblitz	`******`

实体：com.acme.User

子实体：com.acme.UserProfile extends User

扩展字段：picture

user_profiles

id	user_id	picture
1	1	http://...

user_profiles (冗余设计)

id	name	password	picture
1	mercyblitz	`******`	http://...

```
<bean id="user" class="com.acme.User" />
```

```
String className = "com.acme.User";
Class klass = Class.forName(className);
User user = (User) klass.newInstance();
```

复合主键

primary key(id,name)

@IdClass

public class PrimaryKey {

    Long id;

    String name;

}

JPA 组合和关联

JPA 实体管理器

EntityManager == Hibernate Session = MyBatis Session >= JDBC Connection

- 缓存（一级、二级）
- 延迟加载（特有功能）

事务提交

User(id = 1, name = mercyblitz)

TX commit -> users(id,mercyblitz)

User(id = 1 , name = mercyblitz -> xiaomage) (游离相对于数据库)

EntityManager.persit(user) -> 同步数据库

事务回滚

User(id = 1, name = mercyblitz)

EntityManager.persit(user) -> 同步数据库

TX rollback -> users(id,小马哥) (老的数据)

JPA 项目整合

基础知识

最核心的用户 API

- `EntityManager` -> CRUD
  - Hibernate Session 对象
- `EntityTransaction` -> 涉及数据库事务
- `javax.persistence.Query` -> JPA 查询 API 接口（面向对象）

JPA SPI

- Persistence -> API 编程入口
  - Persistence 单元名称 + JPA 实现者属性
- PersistenceProvider -> JPA 厂商实现接口
  - Hibernate -> org.hibernate.jpa.HibernatePersistenceProvider
- PersistenceProviderResolverHolder
- PersistenceUnitInfo
  - 基本的配置信息

        - getPersistenceUnitName()
        - getPersistenceProviderClassName()
        - getTransactionType()
    - 增加字节码提升（ClassTransformer）

获取 EntityManager 基本步骤

1. Persistence API 获取 EntityManagerFactory
    - 通过 META-INF/persistence.xml 文件配置
    - 实现 Persistence
2. EntityManagerFactory 获取 EntityManager

编程模型

- 面向注解
    - 我们对代码有修改权利
- 面向 XML 配置
    - 我们对目标代码没有权利，jar 引入 Entity Class
        - client API

关联技术

Tomcat JNDI 实现与 Spring IoC 的相似度

org.apache.naming.factory.BeanFactory 与 FactoryBean

javax.naming.spi.ObjectFactory 与 Spring ObjectFactory

Spring Data JPA

= Spring Data Commons + Spring JPA

基本用法

```
@Resource
private EntityManager entityManager;

@Autowired
private EntityManager entityManager2;
```

Hibernate

假设有一个 Entity Class 关联其他 Class

```
public class User {

    private List<Address> addresses;

    public List<Address> getAddresses(){ //
Lazy Load（加载）
        ...
    }
}
```

延迟加载是通过类的字节码提升来做的，因此，从 Session 获取到的 Entity Object 的类型是字节码提升的类，那么原始的 Entity 类型是它的父类，所以原始 Entity 类型不能修饰为 final。

Session load 和 get 方法的区别?

get 有缓存

load 重新加载（无缓存）

延迟加载

字节码提升

- CGLIB：支持类代理
- Java 动态代理：仅支持接口
- Javassist：支持类代理，支持字段拦截